Predictive model for credit default via Logistic Regression and Random Forest

Sean Stanislaw

Submitted in partial fulfilment of the requirements for

Professional Certificate in Data Science

Abstract

The goal of credit scoring models is to predict the creditworthiness of a customer and determine whether they will be able to meet a given financial obligation or default on it. Such models allow a financial institution to minimize the risk of loss by setting decision rules regarding which customers receive loan and credit card approvals.

In this capstone project we use Logistic regression and Random Forest to predict the creditworthiness of bank customers using predictors related to their personal status and financial history. Model adequacy and robustness checks are performed to ensure that the model is being properly fitted and interpreted.

Dataset - The source of dataset is Kaggle Credit risk modelling – Case Study

The dataset consists of the following fields:

- **Loan Status:** A categorical variable indicating if the loan was paid back or defaulted.
- **Loan Amount:** This is the loan amount borrowed
- **Home Ownership:** Categorical variable indicating home ownership. Values are "Rent", "Home Mortgage", and "Own". If the value is OWN, then the customer is a home owner with no mortgage.
- **Person Income:** Annual income of the applicant.
- **Person Employment Length:** For how long an individual has been at current Job.
- **Loan Grade:** Credit Score assigned to the application based on Internal credit scoring model.
- **Loan Interest Rate:** Interest rate given by the system to that application.
- **Loan Percent Income:** Ratio of Loan applied to Income of the individual.
- **Default on Credit File:** Weather the individual has any default listed on credit file.
- **Length of Credit File:** The years since the first entry in the customer's credit history
- **Age:** Age of the Applicant.

Research Methodology

Proposed Statistical tools: Logistic regression and Random Forest.

Data cleaning, data exploration and visualization:

- All missing Values to be dropped from the data set row wise
- Cross tabulations will be done on Loan status to identify the number of samples that defaulted to the Number of samples that did not default.
- Cross tabulations will be done on Grades assigned through Internal credit scoring to ascertain proportion of default for every grade.
- Cross tabulations will be done on home status to ascertain how their living situation i.e., renting, owner or mortgagee had a higher rate of default
- The data set will be split in 75% Training and 25% testing.

Logistic Regression:

- 3 Cut off levels will be used to ascertain best performing models 20%, 35% and 50%.
- We will use best AUC (area under curve) to ascertain the best cut-off, along with confusion matrix which basically tabulates each combination of prediction and actual value
- We all test the model for Sensitivity and specificity
- Finally, we will derive the overfall best fit score and compare all 3 to arrive at the best possible cut-off for out model

Data exploration

Our Dataset has 28638 observations of 11 variables after dropping NA variables

1. **Loan Status**
   1 – Did not default
   0 – Defaulted

0	1
22435	6203
0.783	0.217

From the above table we can infer 78.3% of the applicants did not default on their loan while 21.7% applicant defaulted

2. **Risk Grading**

credit_risk_new$loan_grade	credit_risk_new$loan_status 0	1	Total
A	8498	904	9402
	0.904	0.096	0.328
B	7698	1453	9151
	0.841	0.159	0.320
C	4542	1157	5699
	0.797	0.203	0.199
D	1325	1923	3248
	0.408	0.592	0.113
E	308	562	870
	0.354	0.646	0.030
F	63	146	209
	0.301	0.699	0.007
G	1	58	59
	0.017	0.983	0.002
Total	22435	6203	28638

From the above table we can infer that grade A, B, C had the lowest default rate and there is a significant jump from 20.3% default rate for grade C to 59.2% for Grade D

3. **Home Ownership status**

credit_risk_new$person_home_ownership	credit_risk_new$loan_status 0	1	Total
MORTGAGE	10316	1485	11801
	0.874	0.126	0.412
OTHER	67	27	94
	0.713	0.287	0.003
OWN	2046	146	2192
	0.933	0.067	0.077
RENT	10006	4545	14551
	0.688	0.312	0.508
Total	22435	6203	28638

From the above table we can infer Mortgage and Home owner were lower risk category with 87.4% and 93.3% not defaulting on loan.

Introduction - Logistic Regression

Logistic regression is one of the most important models for categorical response data. It is an example of a generalized linear model whose main use is to estimate the probability that a binary response occurs based on a number of predictor variables. Logistic regression is used in a wide variety of applications including biomedical studies, social science research, marketing as well as financial applications. One example of the latter is the use of binary logistic regression models for credit-scoring, that is: modelling the probability that a customer is creditworthy (i.e., able to meet a financial obligation in a timely manner) using a number of predictors.

We fit a binary logistic regression to the data, using the **GLM function** on the training data

The linear predictor of the model we are fitting is given by:

```
Glum (formula = loan_status ~ person_age + person_income + loan_grade +
person_home_ownership + loan_percent_income, family = "binomial", data = train)
```

Next, we assess which variables explain credit worthiness.

```
Coefficients:
                            Estimate Std. Error z value Pr(>|z|)
(Intercept)                -3.49e+00   1.28e-01  -27.31  < 2e-16 ***
person_age                 -3.05e-03   3.41e-03   -0.89     0.37
person_income              -1.19e-05   8.20e-07  -14.52  < 2e-16 ***
loan_gradeB                 3.52e-01   5.82e-02    6.04  1.5e-09 ***
loan_gradeC                 7.28e-01   6.23e-02   11.69  < 2e-16 ***
loan_gradeD                 2.87e+00   6.76e-02   42.43  < 2e-16 ***
loan_gradeE                 3.07e+00   1.07e-01   28.78  < 2e-16 ***
loan_gradeF                 3.36e+00   2.12e-01   15.88  < 2e-16 ***
loan_gradeG                 6.85e+00   1.11e+00    6.15  7.9e-10 ***
person_home_ownershipOTHER  3.63e-01   3.51e-01    1.03     0.30
person_home_ownershipOWN   -1.63e+00   1.25e-01  -13.01  < 2e-16 ***
person_home_ownershipRENT   8.89e-01   4.79e-02   18.57  < 2e-16 ***
loan_percent_income         8.28e+00   2.05e-01   40.41  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Variables with *** is significant at .001 level of significance and explains credit worthiness of an applicant.

To Predict outcome for test data we use the predict function in R.

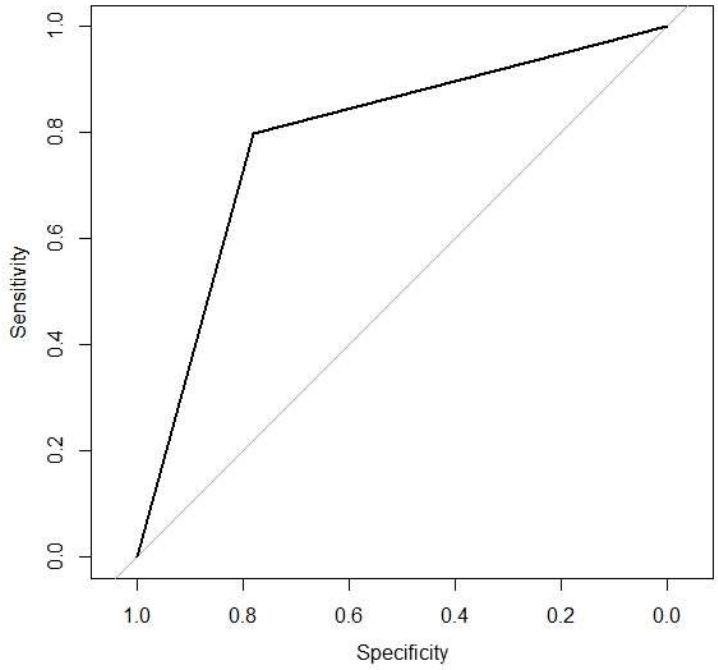
```
pred_log_regression_model <- predict(logit_model_multi, newdata = test, type = "response")
```

Further we check the model for different cut off level 20%, 35% and 50% and plotting their receptive ROC curve.

Model 1 Cut off 50%

Cut off	AUC	sensitivity	Specificity	Mean Model accuracy
50%	0.742	0.88	0.758	0.862

ROC curve (**receiver operating characteristic curve**) for 50% Cut off



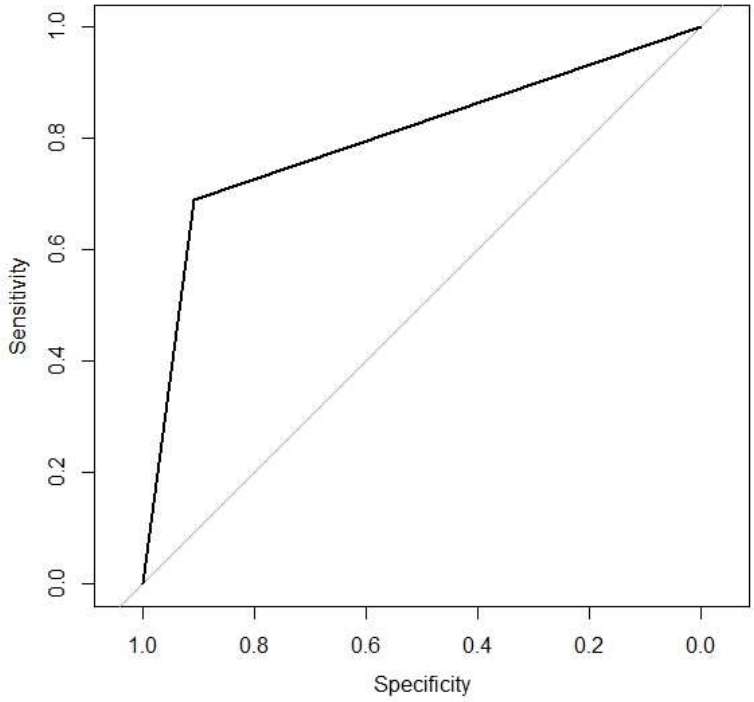
Confusion Matrix plot for 50% Cut off

```
> cf_matrix_3
   class_pred_logit_3
       0    1
0 5346  263
1  727  824
```

Model 2- Cut off 35%

Cut off	AUC	sensitivity	Specificity	Mean Model accuracy
35%	0.8	0.914	0.68	0.862

ROC curve (**receiver operating characteristic curve**) for 35% Cut off



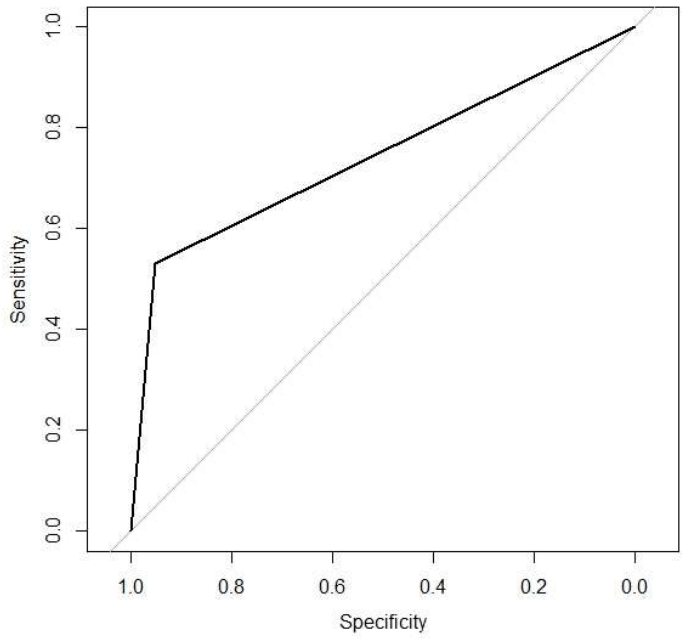
Confusion Matrix plot for 35% Cut off

```
  class_pred_logit_2
     0    1
0 5106  503
1  482 1069
```

Model 3 -20% Cut off

Cut Off	AUC	Sensitivity	Specificity	Mean Model Accuracy
20%	0.79	0.934	0.503	0.786

ROC curve (**receiver operating characteristic curve**) for 20% Cut off



Confusion Matrix plot for 20% Cut off

```
  class_pred_logit_1
      0    1
0 4386 1223
1  312 1239
```

Analysis Summary

Based on the 3 cut off points and various parameters we can conclude 35% performs the bestAs it has the highest AUC of 0.8 and the overall model performance score is 86.25%which is equivalent to 50% cut-off.

Confusion Matrix

For 50% cut off we can observe it has a miss classification rate of 4.6% Credit worthy profile classified as defaulters , whereas it misclassifies 46% Defaulters as credit worthy.

For 35% cut off we can observe it has a miss classification rate of 8.96 % Credit worthy profile classified as defaulters , whereas it misclassifies 31% Defaulters as credit worthy.

Thus, we accept 35% cut off as a better fitting model based on its AUC and Overall performance score along with significant reduction in mis classification of defaulters as credit worthy while only a small increase in Credit worthy being classified as defaulter.

Random Forest – Introduction

Random forests or **Random Decision Forests** are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of overfitting to their training set. Random forests generally outperform decision trees, but their accuracy is lower than gradient boosted trees. However, data characteristics can affect their performance.

The first algorithm for random decision forests was created in 1995 by Tin Kam Housing the random subspace method, which, in Ho's formulation, is a way to implement the "stochastic discrimination" approach to classification proposed by Eugene Kleinberg.

An extension of the algorithm was developed by Leo Breiman and Adele Cutler, who registered "Random Forests" as a trademark in 2006 (as of 2019, owned by Minitab, Inc.). The extension combines Breiman's "bagging" idea and random selection of features, introduced first by Ho and later independently by Amit and Geman in order to construct a collection of decision trees with controlled variance.

Random forests are frequently used as "Blackbox" models in businesses, as they generate reasonable predictions across a wide range of data while requiring little configuration.

Analysis of the Dataset

1. The data set will be split in 75% Training and 25% testing.
2. Multiple models will be tested
3. Problem of Imbalanced dataset to be addressed using ROSE library all methods will be tested

   Over sampling, Under sampling, and both

The random forest package was initialised after correcting for any issues arising due to imbalance dataset.

Model 1 – Number of variables tried for each split was 3

```
Call:
 randomForest(formula = loan_status ~ ., data = train)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 9.1%
Confusion matrix:
      0    1 class.error
0 16497  339      0.0201
1  1616 3026      0.3481
```

Model 2 – All 10 variables were tried at each split

```
Call:
 randomForest(formula = loan_status ~ ., data = train, mtry = 10)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 10

        OOB estimate of  error rate: 8.83%
Confusion matrix:
      0    1 class.error
0 16461  375      0.0223
1  1521 3121      0.3277
```

By analysing OOB estimate and confusion matrix we can ascertain there is probably imbalanced classification problem Imbalanced classifications pose a challenge for predictive modelling as most of the machine learning algorithms used for classification were designed around the assumption of an equal number of examples for each class. This results in models that have poor predictive performance, specifically for the minority class. This is a problem because typically, the minority class is more important and therefore the problem is more sensitive to classification errors for the minority class than the majority class. The distribution of dataset for creditworthy and defaulters is given below which confirm the imbalance in data set with only 21.7% of defaulters included in data set.

0	1
22435	6203
0.783	0.217

We will again estimate the Random Forest model and correcting the imbalance problem using ROSE package under all **3 methods Over Sampling, Under Sampling and Both**.

Model 1 - Method Over Sampling

The following lines of code will be executed

Solve imbalance problem in dataset using over sampling

```
Credit_Risk_Tree_balanced <- ovun.sample(loan_status~ ., data = train, p=0.5,
seed=1,method="over")$data
```

Output:

```
Call:
 randomForest(formula = loan_status ~ ., data = Credit_Risk_Tree_balanced)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 2.14%
Confusion matrix:
      0     1 class.error
0 16258   578     0.03433
1   142 16598     0.00848
```

Model 2 - Method – Both (Over and Under Sampling)

The following lines of code will be executed

Solve imbalance problem in dataset using both, over and under sampling

```
Credit_Risk_Tree_balanced_both <- ovun.sample(loan_status~ ., data = train, p=0.5,
seed=1,method="both")$data
```

Output:

```
Call:
 randomForest(formula = loan_status ~ ., data = Credit_Risk_Tree_balanced_both)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 2.98%
Confusion matrix:
      0     1 class.error
0 10426   370      0.0343
1   271 10411      0.0254
```

Model 3 - Method Under Sampling

Solve imbalance problem in dataset using under sampling

The following lines of code will be executed

```
Credit_Risk_Tree_balanced_under <- ovun.sample(loan_status~ ., data = train,
,method="under",p=0.5, seed=1)$data
```

Output:

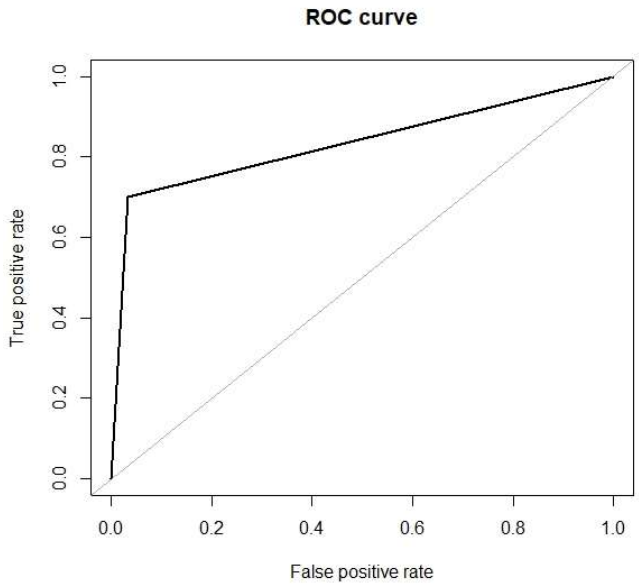
```
Call:
 randomForest(formula = loan_status ~ ., data = Credit_Risk_Tree_balanced_under)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 15.2%
Confusion matrix:
     0    1 class.error
0 4163  447       0.097
1  960 3682       0.207
```

On observing the output, we can conclude that once we have corrected the dataset and transformed it to be balanced it OOB estimate error rate is significantly improved as well as we can observe from the confusion matrix lower mis classification error.

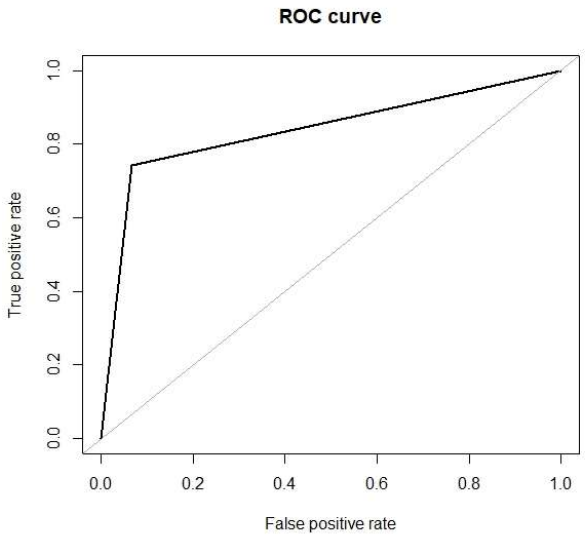
Now we need to **test model accuracy**.

Model 1- Oversampling - ROC curve plot



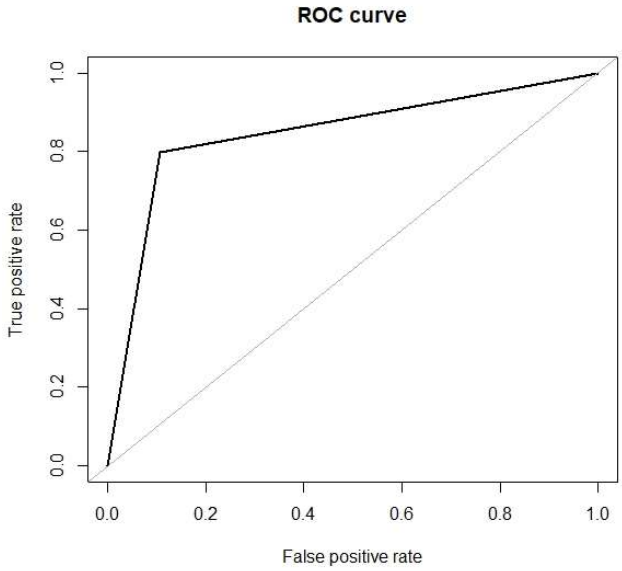
Area under the curve (AUC): 0.834

Model -2 Method – Both (Over and under sampling) ROC curve plot



Area under the curve (AUC): 0.838

Model 3- Method Under sampling



Area under the curve (AUC): 0.845

Result Summary

As we can observe that OOB estimate of error is lowest with Oversampling Method and more or less all 3 models capture similar percentage of Area of under curve (AUC)- Wie will conclude Method 1 is the best fitP model.

References

https://www.datacamp.com/

https://online.stanford.edu/courses/sohs-ystatslearning-statistical-learning

https://stackoverflow.com/

https://towardsdatascience.com/

http://utstat.toronto.edu/~ali/papers/creditworthinessProject.pdf

http://www.endmemo.com/r/ovun.sample.php

https://www.analyticsvidhya.com/

https://en.wikipedia.org/wiki/Random_forest